Multiple choice Inf3705

Chapter 2

1: Which of the following are recognized process flow types?

a. Concurrent process flow
b. Iterative process flow
c. Linear process flow
d. Spiral process flow
e.both a &c

2: Software processes can be constructed out of pre-existing software patterns to best meet the needs of a software project.

a. True
b. False

3: Which of these are standards for assessing software processes?

a. SEI
b. SPICE
c. ISO 9000
d. ISO 9001
E.both b &d

4: The waterfall model of software development is

a. A reasonable approach when requirements are well defined.
b. A good approach when a working program is required quickly.
c. The best approach to use for projects with large development teams.
d. An old fashioned model that is rarely used any more.

5: The incremental model of software development is

a. A reasonable approach when requirements are well defined.
b. A good approach when a working core product is required quickly.
c. The best approach to use for projects with large development teams.
d. A revolutionary model that is not used for commercial products.

6: Evolutionary software process models

a. Are iterative in nature
b. Can easily accommodate product requirements changes
c. Do not generally produce throwaway systems
d. All of the above

7: The prototyping model of software development is

a. A reasonable approach when requirements are well defined.
b. A useful approach when a customer cannot define requirements clearly.
c. The best approach to use for projects with large development teams.
d. A risky model that rarely produces a meaningful product.

8: The spiral model of software development

a. Ends with the delivery of the software product
b. Is more chaotic than the incremental model
c. Includes project risks evaluation during each iteration

d. All of the above

9: The concurrent development model is

a. Another name for concurrent engineering.
b. Defines events that trigger engineering activity state transitions.
c. Only used for development of parallel or distributed systems.
d. Used whenever a large number of change requests are anticipated.

10: The component-based development model is

a. Only appropriate for computer hardware design.
b. Not able to support the development of reusable components.
c. Dependent on object technologies for support.
d. Not cost effective by known quantifiable software metrics.

11: The formal methods model of software development makes use of mathematical methods to

a. Define the specification for computer-based systems
b. Develop defect free computer-based systems
c. Verify the correctness of computer-based systems
d. All of the above

12: Which of these is not one of the phase names defined by the Unified Process model for software development?

a. Inception phase
b. Elaboration phase
c. Construction phase
d. Validation phase

13: Which of these is not a characteristic of Personal Software Process?

a. Emphasizes personal measurement of work product
b. Practitioner requires careful supervision by the project manager
c. Individual practitioner is responsible for estimating and scheduling
d. Practitioner is empowered to control quality of software work products

14: Which of these are objectives of Team Software Process?

a. Accelerate software process improvement
b. Allow better time management by highly trained professionals
c. Build self-directed software teams
d. Show managers how to reduce costs and sustain quality
e. Both b c

15: Process technology tools allow software organizations to compress schedules by skipping unimportant activities.

a. True
b. False

16: It is generally accepted that one cannot have weak software processes and create high quality end products.

a. True
b. False

Chapter 3

1: Agility is nothing more than the ability of a project team to respond rapidly to change.

a. True
b. False

2: Which of the following is not necessary to apply agility to a software process?

a. Eliminate the use of project planning and testing
b. Only essential work products are produced
c. Process allows team to streamline tasks
d. Uses incremental product delivery strategy

3: How do you create agile processes to manage unpredictability?

a. Requirements gathering must be conducted very carefully
b. Risk analysis must be conducted before planning takes place
c. Software increments must be delivered in short time periods
d. Software processes must adapt to changes incrementally
e. Both e and d

4: In agile software processes the highest priorities is to satisfy the customer through early and continuous delivery of valuable software.

a. True
b. False

5: Which of the following traits need to exist among the members of an agile software team?

a. Competence
b. Decision-making ability
c. Mutual trust and respect
d. All of the above

6: In agile development it is more important to build software that meets the customers&#39; needs today than worry about features that might be needed in the future.

a. True
b. False

7: What are the four framework activities found in the Extreme Programming (XP) process model?

a. analysis, design, coding, testing
b. planning, analysis, design, coding
c. planning, analysis, coding, testing
d. planning, design, coding, testing

8: All agile process models conform to a greater or lesser degree to the principles stated in the "Manifesto for Agile Software Development".

a. True
b. False

9: What are the three framework activities for the Adaptive Software Development (ASD) process model?

a. analysis, design, coding
b. feasibility study, functional model iteration, implementation
c. requirements gathering, adaptive cycle planning, iterative development
d. speculation, collaboration, learning

10: Which is not one of the key questions that is answered by each team member at each daily Scrum

meeting?

a. What did you do since the last meeting?
b. What obstacles are you encountering?
c. What is the cause of the problems you are encountering?
d. What do you plan to accomplish be the next team meeting?

11: The Dynamic Systems Development Method (DSDM) suggests a philosophy that is based on the Pareto principle (80% of the application can be delivered in 20% of the time required to build the complete application).

a. True
b. False

12: In Feature Driven Development (FDD) a client-valued feature is a client-valued function that can be delivered in two weeks or less.

a. True
b. False

13: Agile Modeling (AM) provides guidance to practitioner during which of these software tasks?

a. Analysis
b. Design
c. Coding
d. Testing
e. both a and b

14: Agile Unified Process uses the classic UP phased activities (inception, elaboration, construction, transition) to help the team visualize the overall process flow.

a. True
b. False

Chapter 4

1: Software engineering principles have about a three year half-life.
a. True
b. False

2: Which of the following is not one of core principles of software engineering practice?
a. All design should be as simple as possible, but no simpler
b. A software system exists only to provide value to its users.
c. Pareto principle (20% of any product requires 80% of the effort)
d. Remember that you produce others will consume

3: Every communication activity should have a facilitator to make sure that the customer is not allowed to dominate the proceedings.
a. True
b. False

4: The agile view of iterative customer communication and collaboration is applicable to all software engineering practice.
a. True
b. False

5: One reason to involve everyone on the software team in the planning activity is to
a. adjust the granularity of the plan
b. control feature creep
c. get all team members to "sign up" to the plan
d. understand the problem scope

6: Project plans should not be changed once they are adopted by a team.
a. True
b. False

7: Requirements models depict software in which three domains?
a. architecture, interface, component
b. cost, risk, schedule
c. information, function, behavior
d. None of the above

8: The design model should be traceable to the requirements model?
a. True
b. False

9: Teams using agile software practices do not generally create models.
a. True
b. False

10: Which of the following is not one of the principles of good coding?
a. Create unit tests before you begin coding
b. Create a visual layout that aids understanding
c. Refractor the code after you complete the first coding pass
d. Write self-documenting code, not program documentation

11: A successful test I ones that discovers at least one as-yet undiscovered error.
a. True
b. False

12: Which of the following are valid reasons for collecting customer feedback concerning delivered software?

a. Allows developers to make changes to the delivered increment
b. Delivery schedule can be revised to reflect changes
c. Developers can identify changes to incorporate into next increment
d. All of the above

Chapter 5

1: Requirements engineering is a generic process that does not vary from one software project to another.

a. True
b. False

2: During project inception the intent of the of the tasks are to determine

a. basic problem understanding
b. nature of the solution needed
c. people who want a solution
d. none of the above
e. a, b and c

3: Three things that make requirements elicitation difficult are problems of

a. budgeting
b. scope
c. understanding
d. volatility
e. b, c and d

4: A stakeholder is anyone who will purchase the completed software system under development.

a. True
b. False

5: It is relatively common for different customers to propose conflicting requirements, each arguing that his or her version is the right one.

a. True
b. False

6: Which of the following is not one of the context-free questions that would be used during project inception?

a. What will be the economic benefit from a good solution?
b. Who is behind the request for work?
c. Who will pay for the work?
d. Who will use the solution?

7: In collaborative requirements gathering the facilitator

a. arranges the meeting place
b. can not be a customer
c. controls the meeting
d. must be an outsider

8: Which of the following is not one of the requirement classifications used in Quality Function Deployment (QFD)?

a. exciting
b. expected
c. mandatory

d. normal

9: The work products produced during requirement elicitation will vary depending on the

a. size of the budget
b. size of the product being built
c. software process being used
d. stakeholders needs

10: Developers and customers create use-cases to help the software team understand how different classes of end-users will use functions.

a. True
b. False

11: Use-case actors are always people, never system devices.

a. True
b. False

12: The result of the requirements engineering task is an analysis model that defines which of the following problem domain(s)?

a. information
b. functional
c. behavioral
d. all of the above

13: Analysis patterns facilitate the transformation of the analysis model into a design model by suggesting reliable solutions to common problems.

a. True
b. False

14: In win-win negotiation, the customer&#39;s needs are met even though the developer&#39;s need may not be.

a. True
b. False

15: In requirements validation the requirements model is reviewed to ensure its technical feasibility.

a. True
b. False

Chapter 8

1: Which of the following are areas of concern in the design model?

a. architecture
b. data
c. interfaces
d. project scope
e. a,b and c

2: The importance of software design can be summarized in a single word

a. accuracy
b. complexity
c. efficiency
d. quality

3: Which of these are characteristics of a good design?

a. exhibits strong coupling between its modules
b. implements all requirements in the analysis model
c. includes test cases for all components
d. provides a complete picture of the software
e. both b and d

4: Which of the following is not a characteristic common to all design methods?

a. configuration management
b. functional component representation
c. quality assessment guidelines
d. refinement heuristics

5: What types of abstraction are used in software design?

a. control
b. data
c. environmental
d. procedural
e. a, b, and d

6: Which of the following can be used to represent the architectural design of a piece of software?

a. Dynamic models
b. Functional models
c. Structural models
d. All of the above

7: Design patterns are not applicable to the design of object-oriented software?

a. True
b. False

8: Since modularity is an important design goal it is not possible to have too many modules in a proposed design.

a. True
b. False

9: Information hiding makes program maintenance easier by hiding data and procedure from unaffected parts of the program.

a. True
b. False

10: Cohesion is a qualitative indication of the degree to which a module

a. can be written more compactly.
b. focuses on just one thing.
c. is able to complete its function in a timely manner.
d. is connected to other modules and the outside world.

11: Coupling is a qualitative indication of the degree to which a module

a. can be written more compactly.
b. focuses on just one thing.
c. is able to complete its function in a timely manner.
d. is connected to other modules and the outside world.

12: When using structured design methodologies the process of stepwise refinement is unnecessary.

a. True
b. False

13: Software designs are refactored to allow the creation of software that is easier to integrate, easier to test, and easier to maintain.

a. True
b. False

14: Which of the following is not one of the five design class types

a. Business domain classes
b. Entity classes
c. Process classes
d. User interface classes

15: Which design model elements are used to depict a model of information represented from the user&#39;s view?

a. Architectural design elements
b. Component-level design elements
c. Data design elements
d. Interface design elements

16: Which design is equivalent to the floor plan of a house?

a. Architectural design
b. Component-level design
c. Data design
d. Interface design

17: Which design model is equivalent to the detailed drawings of the access points and external utilities for a house?

a. Architectural design
b. Component-level design
c. Data design
d. Interface design

18: Which design model is equivalent to a set of detailed drawings for each room in a house?

a. Architectural design
b. Component-level design
c. Data design
d. Interface design

19: The deployment design elements specify the build order for the software components.

a. True
b. False

Chapter 10

1: In the most general sense a component is a modular building block for computer software.

a. True
b. False

2: In the context of object-oriented software engineering a component contains

a. attributes and operations
b. instances of each class
c. roles for each actor (device or user)
d. set of collaborating classes

3: In traditional software engineering modules must serve in which of the following roles?

a. Control component
b. Infrastructure component
c. Problem domain component
d. All of the above

4: Software engineers always need to cerate components from scratch in order to meet customer expectations fully.

a. True
b. False

5: Which of the following is not one of the four principles used to guide component-level design?

a. Dependency Inversion Principle
b. Interface Segregation Principle
c. Open-Closed Principle
d. Parsimonious Complexity Principle

6: The use of stereotypes can help identify the nature of components at the detailed design level.

a. True
b. False

7: Classes and components that exhibit functional, layer, or communicational cohesion are relatively easy to implement, test, and maintain.

a. True
b. False

8: Software coupling is a sign of poor architectural design and can always be avoided in every system.

a. True
b. False

9: In component design elaboration requires which of the following elements to be describe in detail?

a. Algorithms
b. Attributes
c. Interfaces
d. Operations
e. b, c and d

10: In component-level design persistent data sources refer to

a. Component libraries
b. Databases
c. Files
d. All of the above
e. both b and c

11: WebApp content design at the component level focuses on content objects and the manner in which they interact.

a. True
b. False

12: A WebApp functional architecture describes the key functional components and how they interact with each other.

a. True
b. False

13: Which of these constructs is used in structured programming?

a. branching
b. condition
c. repetition
d. sequence
e. b, c and d

14: Which of these is a graphical notation for depicting procedural detail?

a. box diagram
b. decision table
c. ER diagram
d. flowchart

15: A decision table should be used

a. to document all conditional statements
b. to guide the development of the project management plan
c. only when building an expert system
d. when a complex set of conditions and actions appears in a component

16: A program design language (PDL) is often a

a. combination of programming constructs and narrative text
b. legitimate programming language in its own right
c. machine readable software development language
d. useful way to represent software architecture

17: In component-based software engineering, the development team examines the requirements to see which are amenable to composition, rather than construction, before beginning detailed design tasks.

a. True
b. False

18: Which of the following is not one of the major activities of domain engineering?

a. analysis
b. construction
c. dissemination
d. validation

19: Which of the following factors would not be considered during component qualification?

a. application programming interface (API)
b. development and integration tools required
c. exception handling
d. testing equipment required

20: Which is the following is a technique used for component wrapping?

a. black-box wrapping
b. clear-box wrapping
c. gray-box wrapping

d. white-box wrapping

21: Which of the following is not one of the issues that form a basis for design for reuse?
a. object-oriented programming
b. program templates
c. standard data
d. standard interface protocols

22: In a reuse environment, library queries are often characterized using the ________ element of the 3C Model.
a. concept
b. content
c. context
d. all of the above

Chapter 17

1: In software quality assurance work there is no difference between software verification and software validation.
a. True
b. False

2: The best reason for using Independent software test teams is that
a. software developers do not need to do any testing
b. strangers will test the software mercilessly
c. testers do not get involved with the project until testing begins
d. the conflicts of interest between developers and testers is reduced

3: What is the normal order of activities in which traditional software testing is organized?
a. integration testing
b. system testing
c. unit testing
d. validation testing
e. c, a, d and b

4: By collecting software metrics and making use of existing software reliability models it is possible to develop meaningful guidelines for determining when software testing is done.
a. True
b. False

5: Which of the following strategic issues needs to be addressed in a successful software testing process?
a. conduct formal technical reviewsprior to testing
b. specify requirements in a quantifiable manner
c. use independent test teams
d. wait till code is written prior to writing the test plan
e. both a and b

6: Which of the following need to be assessed during unit testing?
a. algorithmic performance
b. code stability
c. error handling
d. execution paths

e. both a and b

7: Units and stubs are not needed for unit testing because the modules are tested independently of one another.

a. True
b. False

8: Top-down integration testing has as it&#39;s major advantage(s) that

a. low level modules never need testing
b. major decision points are tested early
c. no drivers need to be written
d. no stubs need to be written
e. both b and c

9: Bottom-up integration testing has as it&#39;s major advantage(s) that

a. major decision points are tested early
b. no drivers need to be written
c. no stubs need to be written
d. regression testing is not required

10: Regression testing should be a normal part of integration testing because as a new module is added to the system new

a. control logic is invoked
b. data flow paths are established
c. drivers require testing
d. all of the above
e. both and a and b

11: Smoke testing might best be described as

a. bulletproofing shrink-wrapped software
b. rolling integration testing
c. testing that hides implementation errors
d. unit testing for small programs

12: When testing object-oriented software it is important to test each class operation separately as part of the unit testing process.

a. True
b. False

13: The OO testing integration strategy involves testing

a. groups of classes that collaborate or communicate in some way
b. single operations as they are added to the evolving class implementation
c. operator programs derived from use-case scenarios
d. none of the above

14: Since many WebApps evolve continuously, the testing process must be ongoing as well.

a. True
b. False

15: The focus of validation testing is to uncover places that s user will be able to observe failure of the software to conform to its requirements.

a. True
b. False

16: Software validation is achieved through a series of tests performed by the user once the software is deployed in his or her work environment.

a. True
b. False

17: Configuration reviews are not needed if regression testing has been rigorously applied during software integration.

a. True
b. False

18: Acceptance tests are normally conducted by the

a. developer
b. end users
c. test team
d. systems engineers

19: Recovery testing is a system test that forces the software to fail in a variety of ways and verifies that software is able to continue execution without interruption.

a. True
b. False

20: Security testing attempts to verify that protection mechanisms built into a system protect it from improper penetration.

a. True
b. False

21: Stress testing examines the pressures placed on the user during system use in extreme environments.

a. True
b. False

22: Performance testing is only important for real-time or embedded systems.

a. True
b. False

23: Debugging is not testing, but always occurs as a consequence of testing.

a. True
b. False

24: Which of the following is an approach to debugging?

a. backtracking
b. brute force
c. cause elimination
d. code restructuring
e. a, b, and c

Chapter 18

1: With thorough testing it is possible to remove all defects from a program prior to delivery to the customer.

a. True
b. False

2: Which of the following are characteristics of testable software?

a. observability
b. simplicity
c. stability
d. all of the above

3: The testing technique that requires devising test cases to demonstrate that each program function is operational is called

a. black-box testing
b. glass-box testing
c. grey-box testing
d. white-box testing

4: The testing technique that requires devising test cases to exercise the internal logic of a software module is called

a. behavioral testing
b. black-box testing
c. grey-box testing
d. white-box testing

5: What types of errors are missed by black-box testing and can be uncovered by white-box testing?

a. behavioral errors
b. logic errors
c. performance errors
d. typographical errors
e. both b and d

6: Program flow graphs are identical to program flowcharts.

a. True
b. False

7: The cyclomatic complexity metric provides the designer with information regarding the number of

a. cycles in the program
b. errors in the program
c. independent logic paths in the program
d. statements in the program

8: The cyclomatic complexity of a program can be computed directly from a PDL representation of an algorithm without drawing a program flow graph.

a. True
b. False

9: Condition testing is a control structure testing technique where the criteria used to design test cases is that they

a. rely on basis path testing
b. exercise the logical conditions in a program module

c. select test paths based on the locations and uses of variables
d. focus on testing the validity of loop constructs

10: Data flow testing is a control structure testing technique where the criteria used to design test cases is that they
a. rely on basis path testing
b. exercise the logical conditions in a program module
c. select test paths based on the locations and uses of variables
d. focus on testing the validity of loop constructs

11: Loop testing is a control structure testing technique where the criteria used to design test cases is that they
a. rely basis path testing
b. exercise the logical conditions in a program module
c. select test paths based on the locations and uses of variables
d. focus on testing the validity of loop constructs

12: Black-box testing attempts to find errors in which of the following categories
a. incorrect or missing functions
b. interface errors
c. performance errors
d. none of the above
e. a, b and c

13: Graph-based testing methods can only be used for object-oriented systems
a. True
b. False

14: Equivalence testing divides the input domain into classes of data from which test cases can be derived to reduce the total number of test cases that must be developed.
a. True
b. False

15: Boundary value analysis can only be used to do white-box testing.
a. True
b. False

16: Orthogonal array testing enables the test designer to maximize the coverage of the test cases devised for relatively small input domains.
a. True
b. False

17: Test derived from behavioral class models should be based on the
a. data flow diagram
b. object-relation diagram
c. state transition diagram
d. use-case diagram

18: Client/server architectures cannot be properly tested because network load is highly variable.
a. True
b. False

19: Real-time applications add a new and potentially difficult element to the testing mix
a. performance
b. reliability

c. security
d. time

Chapter 19

1: It is not possible to test object-oriented software without including error discovery techniques applied to the system OOA and OOD models..
a. True
b. False

2: The correctness of the OOA and OOD model is accomplished using formal technical reviews by the software quality assurance team.
a. True
b. False

3: The consistency of object-oriented models may be judged by reviewing the CRC card model.
a. True
b. False

4: Test case design for OO software is driven by the algorithmic detail of the individual operations.
a. True
b. False

5: Integration testing of object-oriented software can be accomplished by which of the following testing strategies?
a. Cluster testing
b. Glass-box testing
c. Thread-based testing
d. Use-based testing
e. a, c, and d

6: Validation of object-oriented software focuses on user visible actions and outputs from the system.
a. True
b. False

7: Encapsulation of attributes and operations inside objects makes it easyto obtain object state information during testing.
a. True
b. False

8: Use-cases can provide useful input into the design of black-box and state-based tests of OO software.
a. True
b. False

9: Fault-based testing is best reserved for
a. conventional software testing
b. operations and classes that are critical or suspect
c. use-case validation
d. white-box testing of operator algorithms

10: Testing OO class operations is made more difficult by
a. encapsulation
b. inheritance
c. polymorphism

d. both b and c

11: Scenario-based testing

a. concentrates on actor and software interaction
b. misses errors in specifications
c. misses errors in subsystem interactions
d. both a and b

12: Deep structure testing is not design to

a. object behaviors
b. communication mechanisms
c. exercise object dependencies
d. exercise structure observable by the user

13: Random order tests are conducted to exercise different class instance life histories.

a. True
b. False

14: Which of these techniques is not useful for partition testing at the class level

a. attribute-based partitioning
b. category-based partitioning
c. equivalence class partitioning
d. state-based partitioning

15: Multiple class testing is too complex to be tested using random test cases.

a. True
b. False